

A Survey: Towards a Robust Deep Neural Network in Text Domain

Wenqi Wang[†], Lina Wang^{†*}, *Member, IEEE*, Run Wang[‡], Aoshuang Ye[†], Benxiao Tang[†],

Abstract—Deep neural networks (DNNs) have gained remarkable success in various tasks. However, they have shown an inherent vulnerability to adversarial examples, which are maliciously crafted by attackers to make target DNNs misbehave. The threat of adversarial examples widely exists in image, text, and so on. Researchers have proposed many methods to solve this problem. Inspired by the works in image, studies on adversarial examples develop rapidly in the text. In this article, we present a comprehensive review of adversarial examples in text, including attack, defense, testing, and verification approaches. First, we introduce the basic knowledge about adversarial examples, like formulation, categorization, and metric measures. Then, we give a taxonomy of recent researches on adversarial examples in text as well as their advantages and shortcomings. Finally, we discuss the challenges in adversarial texts and some research points of this aspect.

Index Terms—Adversarial attack and defense, Adversarial example, Deep neural networks, Testing and verification.

I. INTRODUCTION

Nowadays, DNNs have solved masses of significant problems in various areas, such as computer vision [1], [2], audio [3], [4], and natural language processing (NLP) [5], [6]. Due to their great success, DNN-based systems are widely deployed in the physical world, including some sensitive security areas [7]–[11]. However, Szegedy et al. [12] found that crafted inputs could easily fool DNN-based models. They are called adversarial examples. Their categories are diverse, varying from image to audio or others. It means that most deployed systems based on DNNs are under the potential threat of adversarial attacks. For example, sign recognition system [13], object recognition system [14], audio recognition or control system [15]–[17], and malware detection system [18], [19] are all hard to defend against these attacks.

Many NLP tasks also employ DNN-based models, such as recommendation, sentiment analysis, and question answering systems. Thus, they are also under the threat of adversarial examples. For instance, people are increasingly inclined to search for related comments before shopping, eating, or watching film. The corresponding items with recommendation scores will be given at the same time. The higher the score is, the more likely it is to be accepted by people. These recommendation apps mainly use sentiment analysis with

the previous comments [20]. Hence, attackers could generate adversarial examples based on natural comments. If the crafted data appears on the networks in large quantities, the recommendation scores of some target objects will sharply decrease or increase. The purpose is to smear competitors or make malicious recommendations for shoddy goods. Besides, adversarial examples can also poison the network environment and hinder detection of malicious information [21]–[23]. Therefore, it is significant to know the operating mechanism and formulate robust measures to resist them.

Faced with this situation, researchers treat adversarial examples as a security problem and pay more attention to them [24], [25]. As a result, the studies on adversarial examples have developed rapidly. Here, we first introduce their definitions to make beginners or talented researchers know what they are. With the development of theory and practice, the definitions of adversarial examples [12], [26]–[28] are varied. But they have two cores in common. One is that the perturbations are small. The other is the ability to fool DNN-based models. It naturally raises an idea to explore why adversarial examples exist in DNNs, how they infect the behaviors of models, and how to solve this problem. Hence, it is of great significance to review the rapid development of adversarial attacks and defenses in recent years. There have been several surveys in image [27], [29]–[32], but few in text [33]–[35]. The works in [33], [34] are partly related to adversarial texts. The remaining one [35] compares the methods in the image and describes how adversarial attacks are implemented in the text. But the defense, testing, and verification do not exist in it, nor do the metric measures of adversarial examples.

Specifically, we present a comprehensive survey on adversarial examples in text-domain, aiming at making interested readers have a better understanding of this concept. For beginners, they can know what adversarial examples are, why they exist, and how they work. For readers with some knowledge, we hope they will be inspired by this article. The major contributions of this article are summarized as follows:

- A systematic survey on adversarial attacks and defenses in text-domain is given. We divide the attacks into image-based, importance-based, and optimization-based methods in the classification task. Attacks in other NLP tasks and defense methods are also covered.
- The comparison and analysis of adversarial attacks on public databases are described from some aspects, including success rate and readability. We categorize the testing and verification methods into several groups according to the way they work.
- We present the general observations of adversarial exam-

[†] W. Wang, L. Wang, A. Ye and B. Tang are with Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, China. E-mail: {wangwenqi_001, lnwang, yasfrost, tangbenxiao}@whu.edu.cn

[‡] R. Wang is with Nanyang Technological University, Singapore. E-mail: runwang@yeah.net

* Lina Wang is the corresponding author.

ples and discuss the existing problems and challenges. Some possible ways are given to help interested researchers inspire.

The rest of this article is organized as follows: we first give some related knowledge about adversarial examples in section II. In section III, we review the adversarial attacks for text classification. Attacks on other NLP tasks are in section IV. The researches about defense are introduced in sections V and VI. One of them is on existing defense methods in text. The other is about how to improve the robustness of DNNs from another point. The discussion and conclusion are in sections VII and VIII.

II. PRELIMINARIES

In this section, we describe some related knowledge, including DNNs-based models, explanations of adversarial examples in DNNs, attack types, and scenarios.

A. Deep neural networks

The neural network is a structure composed of multiple neurons. DNNs are capable of learning high-level features with more complexity and abstraction than general neural networks [36]. They are powerful because they can pick up on patterns in many different features of an input by many layers [37]. A DNN model generally includes input layer, hidden layer, softmax layer, and output layer. Each layer is made up of multiple cells. In the text, there are some frequently used DNNs shown in figure 1, such as recurrent neural network (RNN), long-short term memory (LSTM) [38], and gated recurrent unit (GRU) [39]. The major difference between them is the cells in hidden layers. In an RNN, they are recurrent cells. In the LSTM and GRU, they are different memory cells.

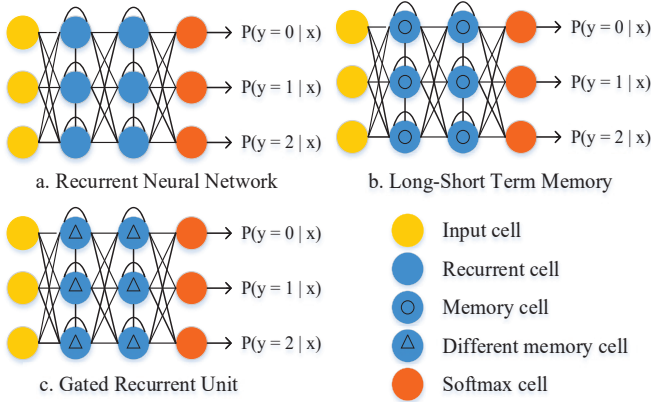


Fig. 1: Some frequently used DNNs

B. Formulation of adversarial examples

The function F of a pre-trained text classification model is to map from an input set to a label set. For a clean text example x , it is correctly classified by F to ground truth label y , where $y \in Y$. Y is the label set of k classes like $\{1, 2, \dots, k\}$. An attacker aims at adding small perturbations in x to generate

adversarial example x' , so that $F(x) = y (y \neq y')$, where $|x - x'| < \delta$. δ is a threshold to limit the size of perturbations. Generally speaking, a good x' should not only fool F , but also be imperceptible to humans, robust to transformations as well as resilient to existing defenses depending on the adversarial goals [40]. Hence, constraint conditions (e.g., semantic similarity) are appended to make x' be indistinguishable from x in some works. Attackers can exploit adversarial examples to cause classification errors.

C. Explanations of the existence

Researchers have been exploring why adversarial examples exist since they were discovered. Nowadays, there are two main points. One is due to the linear structure of DNNs. Its defect will lead to false predictions with a high probability. The other is that adversarial examples are generated based on non-robust features of the data.

Goodfellow et al. [26] explained this question after the emergence of adversarial examples. They thought that the primary cause was the linear nature of DNNs. Although the non-linear activation functions are the main parts of DNNs, but they are linear in a sense. Some of the main activation functions are shown in figure 2, including tanh, sigmoid, relu [41], and ELU¹ [42]. We can see that these functions are very piecewise linear. Besides, there also exist other linear structures, such as the connection of each layer and convolution calculation. A defect of the linear structure is that classification is still possible as it moves away from the decision boundaries, even though there is no training data. However, false predictions are usually made in these places. The presence of adversarial examples may be for this reason.

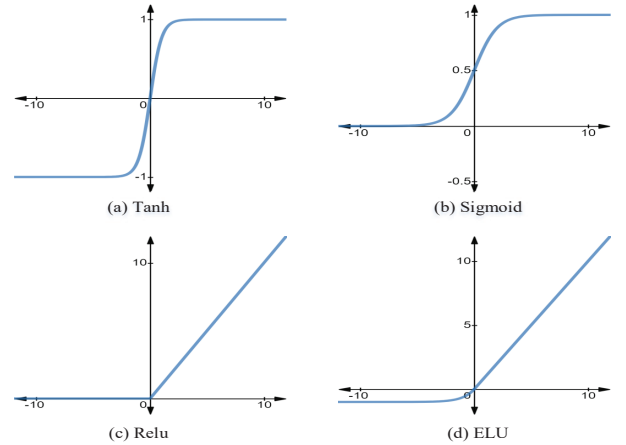


Fig. 2: Some of the main activation functions

But there is a new perspective on the interpretation of adversarial examples. Ilyas et al. [43] claimed, “adversarial vulnerability is a direct result of our models sensitivity to well-generalizing features in the data.” Through a series of

¹Relu and ELU refer to rectified linear units and exponential linear units respectively. ELU is designed for alleviating the vanishing gradient problem of relu.

experiments, they drew the conclusion that adversarial examples are not bugs, but features. The features for prediction can be classified as robust and non-robust, if they exist in standard classification tasks. Both of them are used for predicting the truth label, which is pre-defined. But small perturbations are crafted on non-robust features to make the final prediction incorrect.

D. General classifications of adversarial examples

Figure 3 is a general classifications of adversarial examples on attacks and defenses. The classification information is summarized from the relevant literature, including but not limited to textual aspects.

1) *Taxonomy of adversarial attacks*: It is easy to carry out adversarial attacks on DNNs, even though attackers have little or no knowledge of the target model. Accordingly, attacks are categorized by the authorization level of the model.

Black-box. In a black-box attack, adversaries have no or little knowledge about the target models. In the former scenario, the only thing they know is the final predictions of the models. For example, people only know whether the inputs of a rumor detection system are rumor or non-rumor. Hence, adversaries generally train substitution models and utilize the transferability [26], [44] of adversarial examples to carry out a black-box attack. In the latter scenario, this case can also be called gray-box attacks. Adversaries have little knowledge (e.g., logits²) about the target models. The gray-box attacks are divided into attacks with or without querying. For querying-based methods, adversaries modify the inputs by observing the outputs after each query. Then the process is repeated until adversarial examples are obtained. In another kind of approaches, adversaries can also utilize the transferability of adversarial examples.

White-box. In a white-box attack, adversaries have full access to the target models. They know all about their architectures, parameters, and weights. By the use of the information, adversaries can obtain excellent adversarial examples, which usually perform better than those in black-box attacks.

According to the purpose of adversaries, adversarial attacks are divided into targeted attacks and non-targeted attacks.

Targeted attack. In this case, the generated adversarial example x' is purposefully classified as class t , which is the target of an adversary. It mainly relies on increasing the confidence score (i.e., the logit or the output of softmax layer) of class t .

Non-targeted attack. In this case, the adversary only aims at fooling the model. The result y' can be any class except for y . Contrary to the targeted attack, non-targeted attack operates via reducing the confidence score of the correct class y .

2) *Taxonomy of defenses against adversarial attacks*: There are many reasons for people to study defenses against adversarial examples. There are two main ones below, which are inspired by [45].

- To protect DNN-based systems from adversarial attacks
- To evaluate the robustness of these systems in the worst-case

²Logit is the input of a softmax layer.

It leads to two kinds of directions for defense. One is to operate the data. The general ways are detection and adversarial training. The other is achieved by enhancing the robustness of DNNs, including changing the loss functions of models, testing, and verification methods.

E. Evaluation of adversarial examples

The performance evaluation of adversarial examples is an open-ended question. It reflects the ability to fool DNN-based models. Researchers have used different standards to evaluate their performance. As far as we know, researchers generally evaluate the attacks on target models by accuracy rate or error rate.

- **accuracy rate**: It refers to the ratio of correct discrimination on the inputs. The lower the accuracy rate is, the more effective the adversarial examples are.
- **error rate**: It is the ratio of incorrect discrimination on the inputs. Its use is opposite to the accuracy rate. The higher the error rate is, the more effective the adversarial examples are.

Some researchers prefer to exploit the difference between the accuracy before and after attacks, because it shows the effect more intuitively. These evaluation methods are also used in defenses.

F. Metric

There exists an important issue that the generated adversarial examples should not only fool target models, but also keep the perturbations imperceptible. A good adversarial example must convey the same semantic meaning with the original one, so that metric measures are required to ensure this case. We describe different kinds of measures to evaluate the similarity of adversarial examples in image and text. Then we analyze the reasons why metric measures in the image are not suitable for the text.

1) *Metric measures in image*: In image, most of the recent studies adopt L_p distance to quantify the imperceptibility and similarity between adversarial examples and clean ones. The generalized term for L_p distance is shown in (1):

$$\|\Delta c\|_p = \sqrt[p]{\sum_{i=1}^n |c'_i - c_i|^p} \quad (1)$$

where Δc represents the perturbations. c'_i and c_i are the i -th factors in n -dimensional vectors \vec{c}' and \vec{c} . Equation (1) represents a series of distances, where p could be 0, 2, ∞ , and so on. Specially, when p is equal to zero, $\|\Delta c\|_0 = \sum \text{bool}(c_i \neq 0)$. bool is a logical function. Its result is 0 or 1. L_0 [46]–[48], L_2 [48]–[51], and L_∞ [12], [26], [51]–[54] are the three most frequently used norms in adversarial images.

- L_0 distance evaluates the number of changed pixels before and after modifications. It seems like edit distance, but it does not directly work in text. Results of altered words in text are varied. Some of them are similar to original words and the others may be contrary, even though the L_0 distances of them are equal.

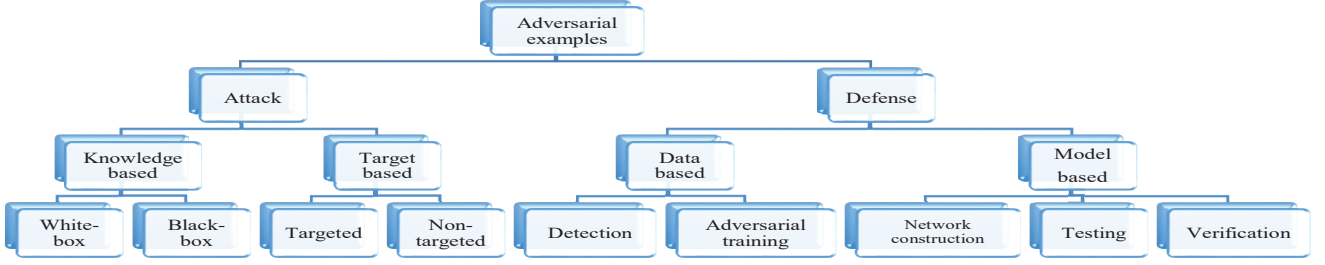


Fig. 3: General categorization of adversarial examples on attacks and defenses

- L_2 represents the Euclidean distance. The original Euclidean distance is the beeline from one point to another in Euclidean space. As the mapping of image, text, or others to this space, it acts as a metric to calculate the similarity between two objects, which are represented as the vectors.
- L_∞ distance measures the maximum change. It is shown in (2):

$$\|\Delta c\|_\infty = \max(|c'_1 - c_1|, \dots, |c'_n - c_n|) \quad (2)$$

\max is a function to choose the maximum factor. Although L_∞ distance is regarded as the optimal distance metric in image, but it may fail in the text. The altered words may not exist in pre-trained dictionary. Hence, they are considered to be unknown words. Their word vectors are also unknown. As a result, the L_∞ distance is hard to calculate.

There are also other metric measures (e.g., structural similarity [55], perturbation sensitivity [56]) which are typical methods for image. Some of them are considered to be more effective than L_p distance, but they can not be directly used in text either. Hence, available metric measures are needed in text, which are different from these in image.

2) *Metric measures in text:* Some measures are proposed to overcome this problem in adversarial texts. We describe some of them which have been demonstrated in the pertinent studies.

Euclidean Distance. For two given word vectors $\vec{m} = (m_1, m_2, \dots, m_k)$ and $\vec{n} = (n_1, n_2, \dots, n_k)$, the Euclidean distance of them is shown in (3):

$$D(\vec{m}, \vec{n}) = \sqrt{(m_1 - n_1)^2 + \dots + (m_k - n_k)^2} \quad (3)$$

where m_i and n_i are the i -th factors in the k -dimensional vectors respectively. The lower the distance is, the more similar they are. Euclidean distance is more used as the metric in image [48]–[51] than in text [57]. It has a generalized term called L_2 norm or L_2 distance.

Cosine Distance. Cosine distance is also a computational method for semantic similarity. It calculates the cosine value of the angle between two vectors. Compared with Euclidean distance, the cosine distance pays more attention to the difference between the directions of two vectors. The more consistent their directions are, the more similar they are. For two given

word vectors \vec{m} and \vec{n} , the cosine similarity of them is shown in (4):

$$D(\vec{m}, \vec{n}) = \frac{\vec{m} \cdot \vec{n}}{\|\vec{m}\| \cdot \|\vec{n}\|} = \frac{\sum_{i=1}^k m_i \times n_i}{\sqrt{\sum_{i=1}^k (m_i)^2} \times \sqrt{\sum_{i=1}^k (n_i)^2}} \quad (4)$$

Jaccard Similarity Coefficient. For two given sets A and B, their Jaccard similarity coefficient $J(A, B)$ is shown in (5):

$$J(A, B) = |A \cap B| / |A \cup B| \quad (5)$$

where $0 \leq J(A, B) \leq 1$. It means that the closer the value of $J(A, B)$ is to 1, the more similar they are. In the text, intersection $A \cap B$ refers to similar words in the samples. Union $A \cup B$ is all words without duplication.

Word Movers Distance (WMD). WMD [58] is a variation of Earth Mover's Distance (EMD) [59]. It can be used to measure the dissimilarity between two text documents, relying on the traveling distance from embedded words of one document to another. Hence, it can quantify the semantic similarity between texts. The lower the value of WMD is, the more similar the two texts are.

Edit Distance. Edit distance is a way to measure the minimum modifications by turning a string to another. The higher it is, the more dissimilar the two strings are. It can be applied to computational biology and natural language processing. Levenshtein distance [60] is also known as edit distance with insertion, deletion, replacement operations in the work of [61].

These metric measures are applied in different situations. Among them, Euclidean distance, cosine distance, and WMD are used on vectors. Adversarial examples and clean ones in text are transformed into vectors. Then these three methods are applied to calculate the similarity between them. On the contrary, Jaccard similarity coefficient and edit distance are directly used on text inputs. They do not need form conversion.

Particularly, Michel et al. [62] proposed a natural criterion for adversarial texts on sequence-to-sequence models. This work focuses on evaluating the semantic equivalence between adversarial examples and clean ones. Experimental results show that strict constraints are useful for keeping meaning-preserving. But whether it is better than the above methods still needs further confirmation.

G. Datasets in Text

To make data more accessible to those who need it, we collect some datasets which have been applied to NLP tasks. Meanwhile, the brief introductions are also given below. These datasets can be downloaded via the corresponding link in the footnote. Table I is the applications of the data. Other datasets used in research works are listed in appendix X.

AG’s News³: It is a set of news with more than one million articles. It is gathered from over 2000 news sources by an academic news search engine named ComeToMyHead. The provided DB version and XML version can be downloaded for any non-commercial use.

DBPedia Ontology⁴: It is a dataset with structured content from the information created in various Wikimedia projects. It has over 68 classes with 2795 different properties. Now there are more than 4 million instances included in this dataset.

Amazon Review⁵: The Amazon review dataset has nearly 35 million reviews spanning Jun 1995 to March 2013, including product and user information, ratings, and a plaintext review. It is collected by over 6 million users in more than 2 million products and categorized into 33 classes with the size ranging from KB to GB.

Yahoo! Answers⁶: The corpus contains 4 million questions and their answers, which can be easily used in the question-answer system. Besides that, a topic classification dataset is also constructed with some main classes.

Yelp Reviews⁷: The provided data is made available by Yelp to enable researchers or students to develop academic projects. It contains 4.7 million user reviews with the types of JSON files and SQL files.

Movie Review (MR)⁸: This is a labeled dataset concerning sentiment polarity, subjective rating and sentences with subjectivity status or polarity. Probably because it is labeled by manual works, the size of this dataset is smaller than others, with a maximum of dozens of MB.

MPQA Opinion Corpus⁹: The Multi-Perspective Question Answering (MPQA) Opinion Corpus is collected from a wide variety of news sources and annotated for opinions or other private states. Three different versions are available to people by the MITRE Corporation. The higher the version is, the richer the contents are.

Internet Movie Database (IMDB)¹⁰: IMDB is crawled from the Internet, including 50000 positive and negative reviews. The average length of the review is nearly 200 words. It is usually used for sentiment classification, including more data than other similar datasets. IMDB also contains the additional unlabeled data, raw text, and already processed data.

SNLI Corpus¹¹: The Stanford Natural Language Inference (SNLI) Corpus is a collection with manually labeled data

TABLE I: Applications of datasets

dataset	application in the work	task
AG’s News	[61], [64]	classification
DBPedia Ontology	[61], [65], [66]	classification
Amazon Review	[61]	classification
Yahoo! Answers	[61]	classification
Yelp Reviews	[61]	classification
Movie Review	[65]–[67]	sentiment analysis
MPQA Opinion Corpus	[66]	classification
IMDB	[57], [65], [67]–[69]	sentiment analysis
SNLI Corpus	[57], [70]	textual entailment, NLI

mainly for natural language inference (NLI) task. There are nearly five hundred thousand sentence pairs written by humans in a grounded context. More details about this corpus can be seen in the work of [63].

III. ADVERSARIAL ATTACKS FOR CLASSIFICATION IN TEXT

The purpose of adversarial attacks is to make DNNs misbehave. They can be regarded as the classification problem in a broad sense, i.e., correct or incorrect judgment. The majority of recent adversarial attacks in text are related to classification tasks. Hence, we first introduce the adversarial attacks with this aspect. In this section, we divide them into two parts based on the desire of attackers. Technical details and corresponding comments of each attack described below are given to make them more clear to readers.

A. Non-targeted attacks for classification

In this part, following studies are all non-targeted attacks. The attackers do not care about the category of misclassification. We have a more detailed division of the attacks, which are image-based, optimization-based, and importance-based methods. The image-based approaches are to apply the methods in the image to the text. The importance-based ones are methods of modifying important words, which highly affect the prediction results.

1) *Image-based approach*: Studies on adversarial examples in image develop faster than those in text. Hence, researchers come up with an idea whether approaches in image can be used in text or not. Some researchers have tried and achieved better results. They propose some efficient approaches based on the fast gradient sign method (FGSM) [26], which needs to operate the gradient.

As far as we know, Papernot et al. [71] is the first to study the problem of adversarial examples in text. They contributed to producing adversarial input sequences on RNN. The authors leveraged computational graph unfolding [72] to evaluate the forward derivative [46], which is related to the embedding inputs of word sequences. The results were then calculated by FGSM to find the adversarial perturbations. However, the corresponding vectors of modified words might not exist. To solve this mapping problem, they set up a specific dictionary to select words, aiming at replacing the original ones. The constraint of this substitution operation was that sign of the difference between substituted and original words was closest

³http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

⁴<https://wiki.dbpedia.org/services-resources/ontology>

⁵<http://snap.stanford.edu/data/web-Amazon.html>

⁶<https://sourceforge.net/projects/yahoodataset/>

⁷<https://www.yelp.com/dataset/download>

⁸<http://www.cs.cornell.edu/people/pabo/movie-review-data/>

⁹<http://mpqa.cs.pitt.edu/>

¹⁰<http://ai.stanford.edu/~amaas/data/sentiment/>

¹¹<https://nlp.stanford.edu/projects/snli/>

to the perturbation. Although their adversarial sequences can make LSTM model misbehave, the words in input sequences are randomly chosen for substitution. The probability of grammatical errors in adversarial examples is very high.

The method proposed by Samanta et al. [68], like adversarial input sequence [71], is also based on FGSM. But the difference between them is the ways to generate adversarial examples. Three modification strategies (i.e., insertion, replacement, and deletion) were introduced, aiming at preserving the semantic meaning of inputs as much as possible. The premise of these modifications was to calculate important or salient words, which deeply affected the results of classification when they were removed. The authors utilized the concept of FGSM to evaluate the importance of a word. The target words were the top k ones with highest contributions, where k was a threshold.

Except for the deletion strategy, both insertion and replacement on top k words required an additional dictionary for replacement. Thus, the authors established a pool of candidates for each word in the experiment, including synonyms, typos, and type-specific keywords. However, it consumes a great deal of time. In the meantime, there may be no candidate pool for the top k words in the actual inputs. This method is also based on importance.

2) *Optimization-based approach*: Different from other methods, Sato et al. [65] operated in the embedding space of inputs and proposed a method named iAdv-Text. The core idea of this method could be seen as an optimization problem. Its main process was to jointly minimize objection function $\mathcal{J}_{iAdvT}(D, W)$ on entire training dataset D with parameters W , which was shown in (6):

$$\mathcal{J}_{iAdvT}(D, W) = \frac{1}{|D|} \arg \min_W \left\{ \sum_{(\hat{X}, \hat{Y}) \in D} \ell(\hat{X}, \hat{Y}, W) + \lambda \sum_{(\hat{X}, \hat{Y}) \in D} \alpha_{iAdvT} \right\} \quad (6)$$

where \hat{X} and \hat{Y} are the inputs and labels respectively. λ is a hyper-parameter to balance the two loss functions. $\ell(\hat{X}, \hat{Y}, W)$ is the loss function of individual training sample (\hat{X}, \hat{Y}) in D .

α_{iAdvT} is a maximization process to find the worst case weights of the direction vectors. Its formula is shown in (7):

$$\alpha_{iAdvT} = \arg \max_{\alpha, \|\alpha\| \leq \epsilon} \left\{ \ell(\vec{w} + \sum_{k=1}^{|V|} a_k d_k, \hat{Y}, W) \right\} \quad (7)$$

where $\sum_{k=1}^{|V|} a_k d_k$ is the perturbation generated from each input on its word embedding vector \vec{w} . ϵ is a hyper-parameter to control adversarial perturbation. a_k is the k -th factor of a $|V|$ -dimensional word embedding vector α . d_k is the k -th factor of a $|V|$ -dimensional direction vector \vec{d} , which is a mapping from one word to another in embedding space. But α_{iAdvT} in (7) was hard to calculate, the authors used (8) instead:

$$\alpha_{iAdvT} = \frac{\epsilon g}{\|g\|_2}, g = \nabla_{\alpha} \ell(\vec{w} + \sum_{k=1}^{|V|} a_k d_k, \hat{Y}, W) \quad (8)$$

iAdv-Text restricts the direction of perturbations to find a substitution. It is in a pre-defined vocabulary rather than an unknown word. Meanwhile, the authors also use cosine similarity to select better perturbations. The readability and semantic similarity are kept.

Similarly, Gong et al. [69] also searched for adversarial perturbations in embedding space. Even though WMD is used by the authors to measure the similarity of clean examples and adversarial examples, the readability of generated adversarial examples seems worse than iAdv-Text.

3) *Importance-based approach*: Unlike previous white-box methods [68], [71], little attention is paid to black-box attacks with adversarial texts. Gao et al. [61] proposed a novel algorithm DeepWordBug in black-box scenario to make DNNs misbehave. A two-stage process was presented by them to generate adversarial perturbations. The first stage was to determine which important tokens to change. The second stage was to create imperceptible perturbations for evading detection. The calculation process for the first stage was shown in (9):

$$CS(x_i) = [F(x_1, \dots, x_{i-1}, x_i) - F(x_1, x_2, \dots, x_{i-1})] + \lambda [F(x_i, x_{i+1}, \dots, x_n) - F(x_{i+1}, \dots, x_n)] \quad (9)$$

where $CS(x_i)$ represents the importance score of the i -th word in the input (x_1, \dots, x_n) . F is a function to evaluate the importance score. λ is a hyper-parameter. Then similar modifications like swap, substitution, deletion, and insertion were applied to manipulate the important tokens. Meanwhile, to preserve the readability of these examples, edit distance was used by the authors.

Li et al. [67] proposed an attack framework TextBugger for generating adversarial examples. It could mislead the deep learning-based text understanding system in both black-box and white-box settings. They followed the similar steps [61] to capture important words and then crafted them. In white-box setting, Jacobian matrix J was used to calculate the importance of each word as follows:

$$C_{x_i} = J_{F(i,y)} = \frac{\partial F_y(x)}{\partial x_i} \quad (10)$$

where $F_y(\cdot)$ represents the confidence value of class y . C_{x_i} is the important score of the i -th word in the input x . The slight changes of words were divided into the character-level and word-level respectively. The ways of modification were insertion, deletion, swap, and substitution. In the black-box setting, the authors segmented documents into sequences. Then they queried the target model to filter out sentences with different predicted labels from the original. The odd sequences were sorted in an inverse order by their confidence score. Then important words were calculated by removing operation in (11):

$$C_{x_i} = F_y(x_1, \dots, x_{i-1}, x_i, x_{i+1}, \dots, x_n) - F_y(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) \quad (11)$$

The final modification process was the same as that in the white-box setting.

B. Targeted attacks for classification

For targeted attack, attackers purposefully control the category of output to be what they desire. The generated examples by them also have similar semantic information with clean ones. This kind of attack is described one by one in the following part.

1) *Image-based approach*: Different from works in [68], [71], Liang et al. [66] first demonstrated that FGSM could not be directly applied in text. Because the input space of text is discrete, image data is continuous. Continuous image has tolerance of tiny perturbations, but the text does not have this kind of feature. Instead, the authors only utilized FGSM to determine what, where, and how to insert, remove, and modify the text input. They conducted two kinds of attacks in different scenarios and used the natural language watermarking [73] technique to make generated adversarial examples compromise their utilities.

In the white-box scenario, the authors defined the conceptions of hot training phrases and hot sample phrases. These two objects were both obtained by leveraging the back-propagation algorithm to compute the cost gradients of examples. The former one shed light on what to insert. The latter one implied where to insert, remove, and modify.

In the black-box scenario, the authors used the idea of fuzzing technique [74] for reference to obtain hot training phrases and hot sample phrases. A core assumption was that the target model could be queried. Examples were fed to target model. Then isometric whitespace was used to substitute origin word each time. The difference between the results before and after modification was the deviation of each word. The larger it was, the more significant the corresponding word was to the classification. Hence, hot training phrases were the most frequent words in a set, which consisted of the largest deviation words for each training sample. And hot sample phrases were the words with largest deviation for every test sample.

2) *Optimization-based approach*: Like one pixel attack [47], a similar method named HotFlip was proposed by Ebrahimi et al. [64]. HotFlip was a white-box attack in text. It relied on an atomic flip operation to swap one token with another based on gradient computation. The authors represented examples as one-hot vectors in the input space. The flip operation was represented by (12):

$$\vec{v}_{ijb} = (\vec{0}, \dots; (\vec{0}, \dots (0, 0, \dots, 0, -1, 0, \dots, 1, 0)_j, \dots, \vec{0})_i; \vec{0}, \dots) \quad (12)$$

The equation (12) means that the j -th character of i -th word in an example is changed from a to b , which are both characters at a -th and b -th places in the alphabet. -1 and 1 in (12) are the corresponding positions for a and b . The change from directional derivative along this vector is calculated to find the biggest increase in the loss $J(x, y)$. The formula is shown in (13):

$$\max \nabla_x J(x, y)^T \cdot \vec{v}_{ijb} = \max_{ijb} \frac{\partial J(b)}{\partial x_{ij}} - \frac{\partial J(a)}{\partial x_{ij}} \quad (13)$$

where x_{ij} is a one-hot vector, which denote the j -th character of i -th word. y refers to the corresponding label vector. T

is a transpose function. HotFlip was used on character-level and word-level by different modifications. Although HotFlip performs well, only a few successful adversarial examples are generated with one or two flips under the strict constraints. Hence, it is not suitable for a large-scale experiment.

A derivative method DISTFLIP was proposed by Gil et al. [75]. They distilled the knowledge of the procedure in a white-box attack (e.g., HotFlip), which was used to train a black-box model. It was a white-to-black process. Through the model, they generated adversarial examples to conduct a black-box attack. This method performs well than HotFlip on a toxicity classifier [76]. Its run-time in generating adversarial examples is ten times faster than HotFlip. But whether this method can distill the knowledge of any white-box attack or not remains to be verified.

Considering the limitation of gradient optimization [23], [64], [71] in black-box case, Alzantot et al. [57] proposed an optimization method based on genetic algorithm [77], [78]. The authors randomly selected words in the input and computed their nearest neighbors by Euclidean Distance in GloVe embedding space [79]. These nearest neighbors were filtered based on language model scores [80] to make them suitable for the surroundings. Hence, only high-ranking words with the highest scores were kept. The substitutions which would maximize the probability of target label were picked from the remaining words. At the same time, above operations were conducted several times to get a generation. If the predicted label of a modified example in a generation was not the target label, two examples were randomly chosen as parents each time to generated the next generation. Then the same process was repeated on the next it. This optimization procedure is done to find a successful attack by genetic algorithm. In this method, random selection words in the sequence to substitute are full of uncertainty. These substitutions may be meaningless, even though the target label is changed.

3) *Summary of adversarial attacks for classification*: These attacks for classification are either popular or representative ones in recent studies. Some main attributes of them are summarized in table II. In table II, the majority of white-box attacks are based on or related to gradient. Gradient-based optimization methods are widely used in image with many variants (e.g., [52], [84]), which can also be applied in text. But they have some shortcomings. They are only used in the white-box scenario. In black-box attack, adversarial attacks mainly rely on the transferability-based or query-based methods. Another limitation is that gradient masking [85] can make them useless in some cases, leading to failure in these methods. Even though this technique is proved to be a failed defense, this kind of methods are not as effective as we think.

Meanwhile, there exist some questions, such as how these attacks perform and which one is better. To make a good comparison on these attacks, we have analyzed the datasets they used. From table I and VII, we can see that IMDB, DBpedia, and MR are three commonly used datasets. But the

¹²<https://iamtrask.github.io/2015/11/15/anyone-can-code-lstm/>

¹³https://github.com/keras-team/keras/blob/master/examples/imdb_lstm.py

¹⁴https://github.com/Smerity/keras_snli/blob/master/snli_rnn.py

TABLE II: Attributes of attacks for classification

Method	White/Black box	Targeted/Non-targeted	Model	Metric	Gradient-related
Alzantot et al. [57]	Black box	Targeted	LSTM ¹³ , RNN ¹⁴	Euclidean Distance	No
DeepWordBug [61]	Black box	Non-targeted	LSTM, char-CNN [81]	Edit Distance	No
HotFlip [64]	White box	Targeted	CNN [82], charCNN-LSTM [83]	Consine Similarity	Yes
iAdv-Text [65]	White box	Non-targeted	LSTM	Consine Similarity	Yes
Text-fool [66]	Both two	Targeted	char-CNN [81]		Yes
TextBugger [67]	Both two	Non-targeted	LR, char-CNN [81], CNN [82]	All metrics except WMD	No
Samanta et al. [68]	White box	Non-targeted	CNN		Yes
Gong et al. [69]	White box	Targeted	CNN	Word Mover Distance(WMD)	Yes
Papernot et al. [71]	White box	Non-targeted	LSTM ¹²		Yes
DISTFLIP [75]	Black box	Non-targeted	GRU		No

TABLE III: Experimental results of adversarial attacks on IMDB and DBPedia

dataset	method	type	black/white	targeted/non-targeted	model	success rate
IMDB	Alzantot et al. [57]	optimization-based	black	targeted	LSTM	97%
	iAdv-Text [65]	optimization-based	white	non-targeted	LSTM	93.92%
	TextBugger [67]	importance-based	white	non-targeted	char-CNN	86.7%
	Samanta et al. [68]	image-based	white	non-targeted	CNN	67.45%
	Gong et al. [69]	optimization-based	white	targeted	CNN	86.66%
DBPedia	DeepWordBug [61]	importance-based	black	non-targeted	LSTM	74.32%
	iAdv-Text [65]	optimization-based	white	non-targeted	LSTM	99.01%
	Text-fool [66]	image-based	white	targeted	char-CNN	84.7%

experimental result of Text-fool [66] on MR is not found. Hence, comparisons are made on IMDB and DBpedia in table III. The results of TextBugger [67] on IMDB and those of Text-fool [66] on DBPedia are in white-box scenario. The success rate of Text-fool is the average. ϵ in the work of [69] is equal to point four. Some attacks are conducted on several models and have both black-box and white-box methods. We only list a good one for comparison.

In table III, white-box attacks are generally better than black-box ones except for the genetic method in Alzantot et al. [57]. This phenomenon may be related to the lack of black-box ones. Some of them are perhaps not worse than the white-box. For example, black-box attacks in TextBugger [67] reach one hundred percent of success rate on several physical systems. In the view of intention, non-targeted attacks usually perform better than targeted ones. The implementation of the former is much simpler than the latter. Besides, the optimization-based methods are much more superior than others. Their higher performances may be due to the stricter constraints than others.

Furthermore, good adversarial examples not only achieve a high success rate to fool the DNN-based models, but also need to have good readability, semantic similarity, and imperceptibility. Hence, we judge through the generated examples in these methods. Instances are shown in table VI, which is in appendix IX. Modifications on the text are generally divided into char-level and word-level. The former operates on the characters, including letters, special symbols, and numbers. The latter modifies the words or sentences. In table VI, the word-level with synonym or suitable sentence seems more imperceptible than the char-level ones, although people are robust against misspelling words [86]. Some char-level methods also perform very well, such as HotFlip. Generally, the more operations there are, the easier it is to be perceived. The more imperceptible the perturbations are, the better the readability and semantic similarity will be.

IV. ADVERSARIAL ATTACKS ON OTHER TASKS

We have reviewed adversarial attacks for classification task above. Next, we solve some other puzzles on adversarial examples, such as what other kinds of tasks or applications can be attacked by adversarial examples, how they are generated in these cases, and whether the crafted examples can be applied in another ways except for attack. The answers to these questions will be described below.

A. Attack on Reading Comprehension Systems

To explore whether reading comprehension systems could understand natural language, Jia et al. [87] inserted adversarial perturbations into paragraphs to test the systems without changing the answers or misleading humans. They extracted nouns and adjectives in the question and replaced them with antonyms. Meanwhile, named entities and numbers were changed by the nearest word in GloVe embedding space [79]. The modified question was transformed into a declarative sentence as the adversarial perturbation, which was then concatenated to the end of the original paragraph. This process is called ADDSENT by the authors. Another way ADDANY randomly chose words of the sentences to craft.

Compared with ADDSENT, ADDANY does not consider the grammaticality of sentences. Meanwhile, it needs to query the model several times. Both two kinds of ways can fool reading comprehension systems well. The core idea is to draw the attention of models on the generated sequences rather than original sequences. Mudrakarta et al. [88] studied adversarial examples on answering question system. But part of their work can strengthen the attacks proposed by Jia et al. [87].

B. Attack on Natural Language Inference Models

Except for reading comprehension systems [87], Minervini et al. [70] put the target on the NLI systems. They cast the generation of adversarial examples as an optimization

problem. Some constraints of First-Order Logic (FOL) are added in NLI to get adversarial examples, which could break these constraints. They maximized the proposed inconsistency loss J_I to search for substitution sets S (i.e., adversarial examples) by using a language model as follows:

$$\begin{aligned} \underset{S}{\text{maximise}} J_I(S) &= [p(S; \text{body}) - p(S; \text{head})]_+, \\ \text{s.t. } \log p_L(S) &\leq \tau \end{aligned} \quad (14)$$

where $[x]_+ = \max(0, x)$. $p_L(S)$ refers to the probability of the sentences in S .

- τ : a threshold on the perplexity of generated sequences
- X_1, \dots, X_n : the set of universally quantified variables in a rule to sequences in S
- $S = \{X_1 \rightarrow s_1, \dots, X_n \rightarrow s_n\}$: a mapping from $\{X_1, \dots, X_n\}$
- $p(S; \text{body})$ and $p(S; \text{head})$: probability of the given rule, after replacing X_i with the corresponding sentence S_i
- body and head : represent the premise and the conclusion of the NLI rules

These generated sequences are used by authors to find the weaknesses of NLI systems.

C. Attack on Neural Machine Translation (NMT)

NMT is another kind of system attacked by adversaries. Belinkov et al. [89] have made some attempts on it. They devised black-box methods depending on natural and synthetic language errors to generate adversarial examples. The naturally occurring errors include typos, misspelling words, or others. Then syntactically adversarial examples are modified by random or keyboard typo types. These experiments are done on three different NMT systems [90], [91]. Experimental results show that these examples could also effectively fool the target systems.

The similar works are also done by Ebrahimi et al. [92]. They conducted an adversarial attack on character-level NMT by employing differentiable string-edit operations. The method of generating adversarial examples is the same in their previous work [64]. Compared with Belinkov et al. [89], the authors demonstrate that adversarial examples in black-box attacks are much weaker than white-box ones in most cases.

D. Attack with Syntactically Controlled Paraphrase Networks (SCPNS)

Iyyer et al. [93] crafted adversarial examples by the use of SCPNS they proposed. They designed this model for generating adversarial examples without decreasing the quality of the input semantics. The general process mainly relied on the encoder-decoder architecture of SCPNS. Given a sequence and a corresponding target syntax structure, the authors encoded them by a bidirectional LSTM model and decoded by the LSTM model. This process was augmented with soft attention over encoded states [94] and the copy mechanism [95]. Then they modified the inputs to the decoder, aiming at incorporating the target syntax structure to generate adversarial examples.

The syntactically adversarial sentences not only can fool pre-trained models, but also improve the robustness of them to syntactic variation. The authors also use crowdsourcing experiment to demonstrate its validity.

V. DEFENSES AGAINST ADVERSARIAL ATTACKS IN TEXT

The constant arms race between adversarial attacks and defenses invalidates conventional wisdom quickly [40]. In fact, defense is more difficult than attack. Few works have been done on this aspect. There are two reasons for this situation. One is that a good theoretical model does not exist for complicated optimization problems like adversarial examples. The other is that tremendous amount of possible inputs may produce the target outputs with a very high possibility. Hence, an actually adaptive defense method is difficult. In this section, we describe some relatively effective methods of defenses against adversarial attacks in text. They are divided into data-based and model-based methods. The former defends by detection or adversarial training. The latter is by enhancing the robustness of models.

A. Adversarial examples detection

Adversarial examples are also a kind of data with a specific purpose. Hence, it is worth considering whether detection is useful against adversarial attacks. Researchers have made various attempts to figure out the differences between adversarial examples and clean ones. Inspired by this view, a series of works [96]–[100] have been conducted to detect adversarial examples and perform relatively well in image. In the text, the ways of modification strategy in some methods will produce misspelling words in generated adversarial examples. This is a distinctly different feature that can be utilized. It naturally comes up with an idea to detect adversarial examples by checking out the misspelling words. Gao et al. [61] used an autocorrector, which was the Python autocorrect 0.3.0 package, to detect the inputs. Li et al. [67] took advantage of a context-aware spelling check service¹⁵ to do the similar work. But experimental results show that this approach is effective on character-level modifications and partly useful on word-level. The differences on modifications strategies between character-level and word-level are the main reasons for the results. This kind of spelling checking method is also not suitable for adversarial examples based on other languages.

Pruthi et al. [101] proposed a word recognition model to detect adversarial examples with misspelling words. The model was applied before a RNN classifier and some backoff strategies were also introduced to achieve a better accuracy. It performs well on the char-level attacks. Although its experiments show that they can also detect word-level attacks, whether it is useful enough for the word-level ones is still unknown. For instance, the example of Alzantot et al. [57] in VI only changes the word "runner" to "racer". It is hard for this method to judge.

Wang et al. [102] proposed a defense method called Synonyms Encoding Method (SEM), which was added before the

¹⁵<https://azure.microsoft.com/zh-cn/services/cognitive-services/spell-check/>

input layer. It was for attacks based on synonyms substitution. The author found the synonymous sentence for an input by clustering and marked them with the same label. Their method did not modify the model or need external knowledge.

B. Adversarial training

Adversarial training [26] is a widely used approach in image [26], [103]. Researchers mix the adversarial examples with original ones as the training dataset to train the model. In text, there are some experiments on adversarial training [61], [64], [67], [104], [105]. However, it fails in the work of [57], mainly because of the different ways of generating adversarial examples. The modifications of the former are insertion, substitution, deletion, and replacement, while the latter makes use of the genetic algorithm to search for adversarial examples.

Over-fitting is another reason why adversarial training method is not always useful and only effective on its corresponding attack. It has been confirmed by Tram'et al. [106] in the image domain, but it remains to be demonstrated in text. Meanwhile, there is a serious problem in adversarial training. When the accuracy of detection on adversarial examples increases, the ability on clean ones decreases. This problem is still under research.

C. Model enhancing against adversarial examples

Except for detection and adversarial training, improving the robustness of the model is another way to resist adversarial examples. Goren et al. [107] formally analyzed, defined, and quantified the notions of robustness of linear learning-to-rank-based relevance ranking function [108], aiming at improving the robustness to small perturbations of documents in the adversarial Web retrieval setting. They adopted the notions of classification robustness [12], [109] to ranking function. Related concepts of pointwise robustness, pairwise robustness, and a variance conjecture were also defined. To quantify the robustness of ranking functions, Kendall's- τ distance [110] and top change were used as normalized measures. Finally, the empirical findings supported the validity of their analyses in two families of ranking functions [111], [112].

Li et al. [113] proposed a method to enhance the robustness of NLI models by the multi-head attention [114]. The authors determined what kind of robustness they wanted. Then the external knowledge (e.g., wordnet [115]) was added to the attention mechanism, which was linked to structured embeddings. Experimental results show that there is a significant improvement on adversarial examples when the knowledge is added to the cross-encoder in their models. Another advantage is that the method does not need extra parameters and any model with attention units can use.

D. Summary of defense methods

Although these methods have achieved better results on their corresponding works, there are also some limitations. The strategy of spell-checking is useless on word-level and sentence-level adversarial examples. Adversarial training exists an over-fitting problem. It will not work when faced with a

new attack method. Model enhancing may be the chief defense strategy in the future, but it is still under exploring. There are many difficulties, such as the choice of loss function and modification of the structure of models. Figure IV is some attributes of these methods, including the efficiency, prospect for development, and expansibility. Except for spell-checking, other two methods have been used in other domains. They also have higher prospects than spell-checking which has great limitations.

TABLE IV: Attributes of defense methods in text

method	expansibility	prospect	application
Spell-checking	low	low	[61], [67], [101]
Adversarial training	high	middle	[57], [61], [64], [67], [104]
Model enhancing	high	high	[107]

VI. TESTING AND VERIFICATION

The current security situation in DNNs seems to fall into a loop. New adversarial attacks are proposed and then followed by new countermeasures which will be subsequently broken [116]. Hence, the formal guarantees for DNNs behavior are badly needed. But it is a hard job. Nobody can ensure that their methods or models are perfect. Recently, what we could do is to make the threat of adversarial attacks as little as possible. The technology of testing and verification helps us deal with the problems from another point. Testing is the use of test cases to discover bugs in the models. Verification is to ensure the normal operation of the models under certain conditions. Both of them are important ways to improve the robustness of DNN-based models.

In this section, we first introduce the methods of testing and verification in image and then are those in text. Even though methods in image have not been applied in text, but their ideas can be used in text like FGSM. We hope someone interested in this aspect can be inspired and comes up with a good defense method used in text or all areas.

A. Testing in image

As increasingly used of DNNs in security-critical domains, it is very significant to have a high degree of trust in the accuracy of models, especially in the presence of adversarial examples. The confidence of the correct behavior of the model is derived from the rigorous testing in various scenarios. More importantly, testing is helpful for understanding the internal behaviors of the network, contributing to the implementation of defense methods. We will survey the testing techniques in DNNs from two aspects. They are testing criteria and test case generation.

1) *Test case generation*: Pei et al. [118] designed a white-box framework DeepXplore to test real-world DNNs with the metric of neuron coverage. They leveraged differential testing to catch the differences of corresponding output between multiple DNNs. The generated adversarial examples have a high neuron coverage. In this way, DeepXplore could trigger the majority logic of the model to find out incorrect behaviors without manual efforts. It performs well in the advanced

TABLE V: Comparison with other four methods

—	DeepConcolic [117]	DeepXplore [118]	DeepGauge [119]	DeepTest [120]	DeepCover [121]
Type of input	normal data or coverage requirements	normal data	normal data	normal data	normal data
Number of input*	single or multiple	multiple	multiple	multiple	multiple
Method of test generation*	concolic	optimisation-based approach	optimization-based approach	greedy search	symbolic execution

* Values of these two items come from the work in [117].

deep learning systems and finds thousands of corner cases which make the systems crash. However, the limitation of DeepXplore is that if all the DNNs make incorrect judgments, it is hard to know where is wrong and how to solve it.

Wicker et al. [122] presented a feature-guided approach to test the resilience of DNNs in black-box scenario against adversarial examples. They treated the process of generating adversarial cases as a two-player turn-based stochastic game with the asymptotic optimal strategy based on Monte Carlo tree search (MCTS) [123] algorithm. In this strategy, there is an idea of reward for accumulating adversarial examples found over the process of gameplay. Via these findings, the authors can evaluate the robustness against adversarial examples.

Besides the feature-guided testing [122], Sun et al. [117] presented DeepConcolic to evaluate the robustness of well-known DNNs, which was the first attempt to apply traditional consoles testing method for these networks. DeepConcolic iteratively used concrete execution and symbolic analysis to generate test suits, aiming at reaching a high coverage and discovering adversarial examples by a robustness oracle. The authors also compared with other testing methods [118]–[121] which were in table V. In terms of input data, DeepConcolic could start with a single input to achieve a better coverage or used coverage requirements as inputs. In terms of performance, DeepConcolic could achieve higher coverage than DeepXplore, but run slower than it.

2) *Testing criteria*: Different from single neuron coverage [118], Ma et al. [119] proposed a multi-granularity testing coverage criteria to measure accuracy and detect erroneous behaviors. They took advantage of four methods [26], [46], [48], [52] to generate adversarial test data to explore the new internal states of the model. The increasing coverage shows that the larger the coverage is, the more possible the defects are to be checked out. Similar work is done by Budnik et al. [124] to explore the output space of the model under test via an adversarial case generation approach.

To solve the limitation of neuron coverage, Kim et al. [125] proposed a new test adequacy criterion called Surprise Adequacy for Deep Learning Systems(SADL) to test DNNs. This method measures the different behaviors between inputs and training data, which is the foundation of the adequacy criterion. Experimental results show that this method can judge whether an input is an adversarial example or not. On the other hand, it also improves the accuracy of DNNs against adversarial examples by retraining.

B. Verification in image

Researchers think that testing is insufficient to guarantee the security of DNNs, especially with unusual inputs like

adversarial examples. Edsger W. Dijkstra once said, “*testing shows the presence, not the absence of bugs*”. Hence, verification techniques on DNNs are needed to study more effective defense methods in adversarial settings.

Pulina et al. [126] are the first to develop a small verification system for a neural network. Since then, related work appears one after another. But verification of the robustness of machine learning models to adversarial examples is still in its infancy [127]. There are only a few researches on related aspects. We group them into three aspects which are search based, global optimization, and over-approximation approach. These works will be introduced in the following part.

1) *Global optimization approach*: Katz et al. [128] presented a novel system named Reluplex to verify DNNs based on Satisfiability Modulo Theory (SMT) [129] solver. They transform the verification into the linear optimization problems with Rectified Linear Unit (ReLU) [41] activation functions. Reluplex is used to find adversarial inputs with the local robustness feature on the ACAS Xu networks [11], but it fails on large networks with the global variant.

For ReLU networks, a part of researches regard the verification as a Mixed Integer Linear Programming (MILP) [130] problem such as Tjeng et al. [131]. They evaluated robustness to adversarial examples from two aspects of minimum adversarial distortion [132] and adversarial test accuracy [133]. Their work is faster than Reluplex with a high adversarial test accuracy, but the same limitation was that it remained a problem to scale it to large networks.

Unlike existing solver-based methods (e.g., SMT), Wang et al. [134] presented ReluVal, which leveraged interval arithmetic [135], to guarantee the correct operations of DNNs in the presence of adversarial examples. They repeatedly partitioned input intervals to find out whether the corresponding output intervals violated security property or not. By contrast, this method is more effective than Reluplex and performs well on finding adversarial inputs.

2) *Search based approach*: Huang et al. [136] proposed a new verification framework which was also based on SMT to verify neural network structures. It relies on discretizing search space and analyzing the output of each layer to search for adversarial perturbations, but the authors find that SMT theory is only suitable for small networks in practice. On the other hand, this framework is limited to many assumptions. Some functions in it are also unclear.

Different from other works, Narodytska et al. [137] verified the secure properties on the binarized neural networks(BNNs) [138]. They leveraged the counterexample-guided search [139] procedure and modern SAT solvers to study the robustness and equivalence of models. The inputs of the models are

judged whether they are adversarial examples or not by two encoding structures *Gen* and *Ver*. It can easily find adversarial examples for up to 95 percent of considered images on the MNIST dataset [140]. But this method works on the middle-sized BNNs rather than large networks.

3) *Over-approximation approach*: Gehr et al. [141] introduced abstract transformers, which could get the outputs of CNN-based models with ReLU. The authors evaluated this approach on a pre-trained defense network [142], aiming at verifying its robustness. Experimental results showed that the FGSM attack could be effectively prevented. They also made some comparisons with Reluplex on both small and large networks. The state-of-the-art Reluplex performed worse than it in the verification of properties and time consumption.

Weng et al. [143] designed two kinds of algorithm to evaluate lower bounds of minimum adversarial distortion, via linear approximations and bounding the local Lipschitz constant. Their methods can be applied to defended networks, especially for adversarial training, to evaluate the effectiveness of them.

C. Testing and verification in text

The methods of testing and verification in text are extremely lacking. There are few works of testing in text. Blohm et al. [144] generated adversarial examples to discover the limitations of their machine reading comprehension model. There are several types of adversarial examples for testing, including word-level and sentence-level in different scenarios [32]. Through testing, the authors found that their model was robust against meaning-preserving lexical substitutions, but failed in importance-based attacks. Besides, experimental results show that some other attributions (e.g., answer by elimination via ranking plausibility [145]) should be added to improve the performance. Cheng et al. [146] proposed a projected gradient method to test the robustness of sequence-to-sequence (seq2seq) models. They found that seq2seq models were more robust to adversarial attacks than CNN-based classifiers. They also addressed the challenges caused by discrete inputs.

Testing in [144] and [146] is different from others in image. It is to judge the robustness of the model by observing its performance to the test cases. However, the similar approaches in image (e.g., DeepXplore [118] and DeepGauge [119]) have considered the neuron coverage of the models. It is worth trying to test DNNs by referring to the methods in image. Besides, testing can be used in other NLP tasks, such as classification, machine translation, and natural language inference.

Jia et al. [147] presented certifiably robust training by optimizing the Interval Bound Propagation (IBP) upper bound [148]. They could limit the loss of the worst-case perturbations and compress the living space of adversarial examples. Their method was provably robust to the attacks with word substitutions on IMDB and SNLI. Like testing, the work of verification is little either. In theory, some methods in image can be applied to text. Because they are for the models themselves, not for the data. For example, methods based on SMT are likely

to be used in text, although they need to be confirmed by experiments.

VII. DISCUSSION

In the previous sections, detailed descriptions of adversarial texts on attack and defense are given to enable readers to have a faster and better understanding of this respect. Next, we present more general observations and challenges on this direction based on the above contents.

A. General observations of adversarial text

Reasons by using misspelled words in some methods:

The motivation by using misspelled words is similar to that in image, which aims at fooling target models with indiscernible perturbations. Some methods tend to conduct character-level modification operations which highly result in misspelled words. Meanwhile, humans are very robust against that case in written language [86].

Transferability in black-box scenario: When the adversaries have no access including querying to the target models, they train a substitute model and utilize the transferability of adversarial examples. Szegedy et al. [12] first found that adversarial examples generated from a neural network could also make another model misbehave by different datasets. It reflects the transferability of the adversarial example. As a result, adversarial examples generated in the substitute model are used to attack the target models, while models and datasets are all inaccessible. Apart from that, the construction adversarial examples with high transferability is a prerequisite to evaluate the effectiveness of black-box attacks. It is also a key metric to evaluate generalized attacks [149]. All the adversarial examples are facing this challenge.

B. Existing problems

Difficulties of adversarial attacks and defenses: There are many reasons for the difficulties of adversarial attacks and defenses. One of the main reasons is that there is not a straightforward way to evaluate proposed works no matter attack or defense. Namely, the convincing benchmarks do not exist in recent works. One good performed attack in a scenario may fail in another, or new defense will soon be defeated in the way beyond the anticipation of defenders. Even though some works are provably sound, rigorous theoretical supports are still needed to deal with the problem of adversarial examples.

Problems on evaluating the performance of attack or defense: Recently, most of researches evaluate their performances of adversarial attacks by success rate or accuracy. Only a few works [61], [67] take scale and efficiency into consideration, although they only list the time spent on the attacks. A question of whether there is a relationship among the scale of dataset, time consumed, and success rate of adversarial attacks is still unknown. If there exists such a relationship, the trade-off of these three aspects may be a research point in the future work. The similar situation also exists in defense.

C. Challenges

Without a universal approach to generate adversarial examples: Because the application of adversarial examples in text rose as a frontier in recent years, the methods of adversarial attacks were relatively few, let alone defenses. Another reason why this kind of method does not exist is the language problem. Almost all recent methods use English datasets. The generated adversarial examples will be useless to the systems with Chinese [150] or other language datasets. Thus, there is not a universal approach to generate adversarial examples. But in our observations, many methods mainly follow a two-step process to generate adversarial examples. The first step is to find important words which have significant impact on the results of classification. Then homologous modifications are used to get adversarial examples.

The lack of a benchmark and toolbox for textual adversarial examples researches: Various methods have been proposed to study adversarial attacks and defenses in text, but there is not a benchmark for them. Researchers use different datasets (in II-G) in their works, resulting in the difficulty of comparing the advantages and disadvantages of these methods. Meanwhile, it also affects the selection of metric measures. Currently, there do not have an exact statement that which metric measure is better in a situation and why it is more useful than others. Some comparisons have been made in Textbugger [67] with several metric measures. The best one in this work may be only suitable for it, but ineffective in other works.

On the other hand, it is also short of an open-source toolbox (e.g., AdvBox [151] and cleverhans [152] in image) for textual adversarial examples researches. The toolboxes in image integrate existing representative methods of generating adversarial images. People can easily do some further studies by them, which reduce time consumption for repetition and promote the development of research in this field. Compared with those in the image, there is only one visual analytics framework proposed by Laughlin et al. [153] in text. But it is lack of diverse attack and defense methods, which can be integrated to make it more powerful.

VIII. CONCLUSION AND FUTURE DIRECTIONS

This article presents a survey about adversarial attacks and defenses on DNNs in text. Even though DNNs have a high performance on a wide variety of NLP, they are inherently vulnerable to adversarial examples, which lead to a high degree of concern about it. This article integrates almost existing adversarial attacks and some defenses focusing on recent works in the literature. The threat of adversarial attacks is real, but defense methods are few. Most existing works have their own limitations, such as application scene, constraint condition, and problems of the method itself. More attention should be paid to the problem of adversarial example, which remains an open issue for designing considerably robust models against adversarial attacks.

Further researches that we can do on adversarial examples are as follows: As an attacker, designing universal perturbations to catch better adversarial examples can be taken into consideration as it works in image [49]. A universal adversarial

perturbation on any text is able to make a model misbehave with high probability. Moreover, more wonderful universal perturbations can fool multi-models or any model on any text. On the other hand, the work of enhancing the transferability of adversarial examples is meaningful in more practical back-box attacks. Besides, the combination of optimization-based and transferability-based methods is another viable way like the work in [154]. On the contrary, defenders prefer to completely revamp this vulnerability in DNNs. But it is no less difficult than redesigning a network. Both of them are long and arduous tasks with the common efforts of many people. At the moment, the defender can draw on methods from image area to text for improving the robustness of DNNs, e.g., adversarial training [142], adding extra layer [155], optimizing cross-entropy function [156], [157], or weakening the transferability of adversarial examples.

ACKNOWLEDGMENTS

This work was partly supported by the National Key R&D Program of China under No. 2016YFB0801100, NSFC under No. 61876134, U1536204 and U1836112.

REFERENCES

- [1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS'12 Proceedings of the 25th International Conference on Neural Information Processing Systems*, vol. 1, 2012, pp. 1097–1105.
- [2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems*, vol. 1, 2015, pp. 91–99.
- [3] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. rahman Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.
- [4] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," 2016, arXiv preprint arXiv:1609.03499.
- [5] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems*, 2014, p. 31043112.
- [6] H. Xu, M. Dong, D. Zhu, A. Kotov, A. I. Carcone, and S. Naar-King, "Text classification with topic-based word embedding and convolutional neural networks," in *BCB '16 Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, 2016, pp. 88–97.
- [7] D. Tao, X. Li, S. Maybank, and X. Wu, "Human carrying status in visual surveillance," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 1670–1677.
- [8] F. Zhang, P. P. K. Chan, B. Biggio, D. S. Yeung, and F. Roli, "Adversarial feature selection against evasion attacks," *IEEE Transactions on Cybernetics*, vol. 46, no. 3, pp. 766–777, 2015.
- [9] J. Saxe and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features," in *Proceedings of the 10th International Conference on Malicious and Unwanted Software (MALWARE)*. IEEE, 2015.
- [10] Z. Yuan, Y. Lu, Z. Wang, and Y. Xue, "Droid-sec: deep learning in android malware detection," in *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM)*, 2014, pp. 371–372.
- [11] K. D. Julian, J. Lopez, J. S. Brush, M. P. Owen, and M. J. Kochenderfer, "Policy compression for aircraft collision avoidance systems," in *Proceedings of the 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, 2016, pp. 1–10.

- [12] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *Proceedings of the International Conference on Learning Representations*, 2014.
- [13] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning models," in the *31th IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
- [14] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, "Adversarial examples for semantic segmentation and object detection," in *IEEE International Conference on Computer Vision*, 2017, pp. 1378–1387.
- [15] R. Taori, A. Kamsetty, B. Chu, and N. Vemuri, "Targeted adversarial examples for black box audio systems," 2018, arXiv preprint arXiv: 1805.07820.
- [16] N. Carlini and D. Wagner, "Audio adversarial examples: Targeted attacks on speech-to-text," in *IEEE Security and Privacy Workshops*. IEEE, 2018.
- [17] H. Yakura and J. Sakuma, "Robust audio adversarial example for a physical attack," 2018, arXiv preprint arXiv: 1810.11793.
- [18] X. Liu, Y. Lin, H. Li, and J. Zhang, "Adversarial examples: Attacks on machine learning-based malware visualization detection methods," 2018, arXiv preprint arXiv:1808.01546.
- [19] W. He and Y. Tan, "Generating adversarial malware examples for black-box attacks based on gan," 2017, arXiv preprint arXiv: 1702.05983.
- [20] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093–1113, 2014.
- [21] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang, "Abusive language detection in online user content," in *WWW '16 Proceedings of the 25th International Conference on World Wide Web*, 2016, pp. 145–153.
- [22] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," in *Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, 2015.
- [23] X. Xiao, B. Yang, and Z. Kang, "A gradient tree boosting based approach to rumor detecting on sina weibo," 2018, arXiv preprint arXiv: 1806.06326.
- [24] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, 2017, p. 506519.
- [25] S. Shen, R. Furuta, T. Yamasaki, and K. Aizawa, "Fooling neural networks in face attractiveness evaluation: Adversarial examples with high attractiveness score but low subjective score," in *IEEE Third International Conference on Multimedia Big Data*. IEEE, 2017.
- [26] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proceedings of the International Conference on Learning Representations*, 2015.
- [27] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *IEEE Access*, vol. 6, pp. 14 410 – 14 430, 2018.
- [28] A. P. Norton and Y. Qi, "Adversarial-playground: A visualization suite showing how adversarial examples fool deep learning," in *IEEE Symposium on Visualization for Cyber Security*, 2017, pp. 1–4.
- [29] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recognition*, vol. 84, pp. 317–331, 2018.
- [30] J. Gilmer, R. P. Adams, I. Goodfellow, D. Andersen, and G. E. Dahl, "Motivating the rules of the game for adversarial example research," arXiv preprint arXiv:1807.06732, 2018.
- [31] Q. Liu, P. Li, W. Zhao, W. Cai, S. Yu, and V. C. M. Leung, "A survey on security threats and defensive techniques of machine learning: A data driven view," *IEEE Access*, vol. 6, pp. 12 103 – 12 117, 2018.
- [32] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," pp. 1–20, 2019.
- [33] H. Xu, Y. Ma, H. Liu, D. Deb, H. Liu, J. Tang, and A. Jain, "Adversarial attacks and defenses in images, graphs and text: A review," arXiv preprint arXiv:1909.08072, 2019.
- [34] Y. Belinkov and J. Glass, "Analysis methods in neural language processing: A survey," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 49–72, 2019.
- [35] W. E. Zhang, Q. Z. Sheng, A. Alhazmi, and C. Li, "Adversarial attacks on deep learning models in natural language processing: A survey," arXiv preprint arXiv:1901.06796, 2019.
- [36] V. Sze, Y.-H. Chen, T.-J. Yang, and J. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295 – 2329, 2017.
- [37] D. Heaven, "Why deep-learning ais are so easy to fool," *Nature*, vol. 574, pp. 163–166, 2019.
- [38] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [39] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," in *Proceedings of Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-8)*, 2014.
- [40] X. Ling, S. Ji, J. Zou, J. Wang, C. Wu, B. Li, and T. Wang, "Deepsec: A uniform platform for security analysis of deep learning model," in *IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 381–398.
- [41] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning*, 2010, pp. 807–814.
- [42] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," in *Proceedings of the International Conference on Learning Representations*, 2016.
- [43] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, "Adversarial examples are not bugs, they are features," arXiv preprint arXiv:1905.02175, 2019.
- [44] M. Naseer, S. H. Khan, S. Rahman, and F. Porikli, "Distorting neural representations to generate highly transferable adversarial examples," 2018, arXiv preprint arXiv: 1811.09020.
- [45] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin, "On evaluating adversarial robustness," 2019, arXiv preprint arXiv: 1902.06705.
- [46] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *IEEE European Symposium on Security and Privacy*. IEEE, 2016.
- [47] J. Su, D. V. Vargas, and S. Kouichi, "One pixel attack for fooling deep neural networks," 2017, arXiv preprint arXiv:1710.08864.
- [48] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy (SP)*, 2017.
- [49] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [50] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2574–2582.
- [51] M. Cisse, Y. Adi, N. Neverova, and J. Keshet, "Houdini: Fooling deep structured prediction models," 2017, arXiv preprint arXiv: 1707.05373.
- [52] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Proceedings of the International Conference on Learning Representations*, 2017.
- [53] S. Sarkar, A. Bansal, U. Mahbub, and R. Chellappa, "Upset and angry: Breaking high performance image classifiers," 2017, arXiv preprint arXiv: 1707.01159.
- [54] S. Baluja and I. Fischer, "Adversarial transformation networks: Learning to generate adversarial examples," 2017, arXiv preprint arXiv: 1703.09387.
- [55] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," vol. 13, no. 4, pp. 600–612, 2004.
- [56] B. Luo, Y. Liu, L. Wei, and Q. Xu, "Towards imperceptible and robust adversarial example attacks against neural networks," in *Proceedings of Association for the Advancement of Artificial Intelligence*, 2018.
- [57] M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. Srivastava, and K.-W. Chang, "Generating natural language adversarial examples," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.
- [58] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger, "From word embeddings to document distances," in *Proceedings of the International Conference on International Conference on Machine Learning*, 2015, pp. 957–966.
- [59] Y. Rubner, C. Tomasi, and L. J. Guibas, "A metric for distributions with applications to image databases," in *ICCV '98 Proceedings of the Sixth International Conference on Computer Vision*. IEEE, 1998, pp. 59–66.
- [60] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," vol. 10, no. 8, pp. 707–710, 1966.

- [61] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, "Black-box generation of adversarial text sequences to evade deep learning classifiers," in *IEEE Security and Privacy Workshops (SPW)*. IEEE, 2018.
- [62] P. Michel, X. Li, G. Neubig, and J. M. Pino, "On evaluation of adversarial perturbations for sequence-to-sequence models," in *Proceedings of the 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.
- [63] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.
- [64] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "Hotflip: White-box adversarial examples for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. ACL, 2018, pp. 31–36.
- [65] M. Sato, J. Suzuki, H. Shindo, and Y. Matsumoto, "Interpretable adversarial perturbation in input embedding space for text," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.
- [66] B. Liang, H. Li, M. Su, P. Bian, X. Li, and W. Shi, "Deep text classification can be fooled," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, 2018, p. 42084215.
- [67] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "Textbugger: Generating adversarial text against real-world applications," in *the Network and Distributed System Security Symposium*, 2019.
- [68] S. Samanta and S. Mehta, "Towards crafting text adversarial samples," 2017, arXiv preprint arXiv:1707.02812.
- [69] Z. Gong, W. Wang, B. Li, D. Song, and W.-S. Ku, "Adversarial texts with gradient methods," 2018, arXiv preprint arXiv:1801.07175.
- [70] P. Minervini and S. Riedel, "Adversarially regularising neural nli models to integrate logical background knowledge," in *the SIGNLL Conference on Computational Natural Language Learning*, 2018.
- [71] N. Papernot, P. McDaniel, A. Swami, and R. Harang, "Crafting adversarial input sequences for recurrent neural networks," in *IEEE Military Communications Conference*, 2016, p. 4954.
- [72] P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural Networks*, vol. 1, no. 4, pp. 339–356, 1988.
- [73] M. Atallah, V. Raskin, M. Crogan, C. Hempelmann, F. Kerschbaum, D. Mohamed, and S. Naik, "Natural language watermarking: Design, analysis, and a proof-of-concept implementation," in *Information Hiding*, p. 185200, 2001.
- [74] M. Sutton, A. Greene, and P. Amini, *Fuzzing: Brute Force Vulnerability Discovery*. Addison-Wesley Professional, 2007.
- [75] Y. Gil, Y. Chai, O. Gorodissky, and J. Berant, "White-to-black: Efficient distillation of black-box adversarial attacks," in *Proceedings of the 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.
- [76] H. Hosseini, S. Kannan, B. Zhang, and R. Poovendran, "Deceiving google's perspective api built for detecting toxic comments," in *Proceedings of the Bright and Dark Sides of Computer Vision: Challenges and Opportunities for Privacy and Security workshop*, 2017.
- [77] E. J. Anderson and M. C. Ferris, "Genetic algorithms for combinatorial optimization: the assemble line balancing problem," *ORSA Journal on Computing*, vol. 6, no. 2, p. 161173, 1994.
- [78] H. Mhlenbein, "Parallel genetic algorithms, population genetics and combinatorial optimization," in *Proceedings of the third international conference on Genetic algorithms*, 1989, pp. 416–421.
- [79] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, p. 15321543.
- [80] C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson, "One billion word benchmark for measuring progress in statistical language modeling," *Computer Science*, 2013.
- [81] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, p. 649657.
- [82] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014.
- [83] Y. Kim, Y. Jernite, D. Sontag, and A. der M Rush, "Character-aware neural language models," in *Proceedings of AAAI*, 2016.
- [84] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," 2017, arXiv preprint arXiv:1710.06081.
- [85] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," 2018, arXiv preprint arXiv:1802.00420.
- [86] R. Graham, "The significance of letter position in word recognition," vol. 22, no. 1, pp. 26–27, 2007.
- [87] R. Jia and P. Liang, "Adversarial examples for evaluating reading comprehension systems," in *Proceedings of the 2017 conference on empirical methods in natural language processing (EMNLP)*, 2017, p. 20212031.
- [88] P. K. Mudrakarta, A. Taly, M. Sundararajan, and K. Dhamdhere, "Did the model understand the question?" in *the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.
- [89] Y. Belinkov and Y. Bisk, "Synthetic and natural noise both break neural machine translation," in *Proceedings of the International Conference on Learning Representations*, 2018.
- [90] J. Lee, K. Cho, and T. Hofmann, "Fully character-level neural machine translation without explicit segmentation," in *Transactions of the Association for Computational Linguistics (TACL)*, 2017.
- [91] R. Sennrich, O. Firat, K. Cho, A. Birch, B. Haddow, J. Hirschler, M. Junczys-Dowmunt, S. Laubli, A. V. M. Barone, J. Mokry, and M. Ndejde, "Nematus: a toolkit for neural machine translation," in *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 2017, p. 6568.
- [92] J. Ebrahimi, D. Lowd, and D. Dou, "On adversarial examples for character-level neural machine translation," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018.
- [93] M. Iyyer, J. Wieting, K. Gimpel, and L. Zettlemoyer, "Adversarial example generation with syntactically controlled paraphrase networks," in *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.
- [94] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proceedings of the International Conference on Learning Representations*, 2014.
- [95] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proceedings of the Association for Computational Linguistics*, 2017.
- [96] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On detecting adversarial perturbations," in *Proceedings of the International Conference on Learning Representations*, 2017.
- [97] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," in *Proceedings of Network and Distributed Systems Security Symposium (NDSS)*, 2018.
- [98] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner, "Detecting adversarial samples from artifacts," 2017, arXiv preprint arXiv:1703.00410.
- [99] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel, "On the (statistical) detection of adversarial examples," 2017, arXiv preprint arXiv:1702.06280.
- [100] K. Roth, Y. Kilcher, and T. Hofmann, "The odds are odd: A statistical test for detecting adversarial examples," 2019, arXiv preprint arXiv:1902.04818.
- [101] D. Pruthi, B. Dhingra, and Z. C. Lipton, "Combating adversarial misspellings with robust word recognition," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.
- [102] X. Wang, H. Jin, and K. He, "Natural language adversarial attacks and defenses in word level," *arXiv preprint arXiv:1909.06723*, 2019.
- [103] C. K. Mummadi, T. Brox, and J. H. Metzen, "Defending against universal perturbations with shared adversarial training," 2018, arXiv preprint arXiv:1812.03705.
- [104] T. Miyato, A. M. Dai, and I. Goodfellow, "Adversarial training methods for semi-supervised text classification," in *Proceedings of the International Conference on Learning Representations*, 2017.
- [105] Y. Cheng, L. Jiang, and W. Macherey, "Robust neural machine translation with doubly adversarial inputs," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. ACL, 2019.
- [106] F. Tram'er, A. Kurakin, N. Papernot, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *Proceedings of the International Conference on Learning Representations*, 2018.
- [107] G. Goren, O. Kurland, M. Tennenholtz, and F. Raiber, "Ranking robustness under adversarial document manipulations," in *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2018.
- [108] T.-Y. Liu, *Learning to Rank for Information Retrieval*. Springer, 2011.
- [109] A. Fawzi, O. Fawzi, and P. Frossard, "Analysis of classifiers' robustness to adversarial perturbations," vol. 107, no. 3, p. 481508, 2018.

- [110] G. S. Shieh, "A weighted kendalls tau statistic," vol. 39, no. 1, pp. 17–24, 1998.
- [111] T. Joachims, "Training linear svms in linear time," in *KDD'06 Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 217–226.
- [112] Q. Wu, C. J. C. Burges, K. M. Svore, and J. Gao, "Adapting boosting for information retrieval measures," vol. 13, no. 3, p. 254270, 2010.
- [113] A. H. Li and A. Sethy, "Knowledge enhanced attention for robust natural language inference," *arXiv preprint arXiv:1909.00102*, 2019.
- [114] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, p. 59986008, 2017.
- [115] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the Association for Computing Machinery*, vol. 38, no. 11, pp. 39–41, 1995.
- [116] L. Ma, F. Juefei-Xu, M. Xue, Q. Hu, S. Chen, B. Li, Y. Liu, J. Zhao, J. Yin, and S. See, "Secure deep learning engineering: A software quality assurance perspective," 2018, arXiv preprint arXiv: 1810.04538.
- [117] Y. Sun, M. Wu, W. Ruan, X. Huang, M. Kwiatkowska, and D. Kroening, "Concolic testing for deep neural networks," in *Proceedings of 33rd ACM/IEEE International Conference on Automated Software Engineering*, 2018.
- [118] K. Pei, Y. Cao, J. Yang, and S. Jana, "Deepxplore: Automated whitebox testing of deep learning systems," in *Proceedings of ACM Symposium on Operating Systems Principles*. ACM, 2017.
- [119] L. Ma, F. Juefei-Xu, J. Sun, C. Chen, T. Su, F. Zhang, M. Xue, B. Li, L. Li, Y. Liu, J. Zhao, and Y. Wang, "Deepgauge: Comprehensive and multi-granularity testing criteria for gauging the robustness of deep learning systems," in *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, 2018.
- [120] Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest: Automated testing of deep-neural-network-driven autonomous cars," in *Proceedings of the 40th International Conference on Software Engineering*. ACM, 2018, pp. 303–314.
- [121] Y. Sun, X. Huang, and D. Kroening, "Testing deep neural networks," 2018, arXiv preprint arXiv: 1803.04792.
- [122] M. Wicker, X. Huang, and M. Kwiatkowska, "Feature-guided black-box safety testing of deep neural networks," in *Proceedings of the International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 2018, pp. 428–426.
- [123] G. M. J.-B. Chaslot, M. H. M. Winands, H. J. van den Herik, J. W. UITERWIJK, and B. Bouzy, "Progressive strategies for monte-carlo tree search," *New Mathematics and Natural Computation*, vol. 4, no. 3, pp. 343–359, 2008.
- [124] C. Budnik, M. Gario, G. Markov, and Z. Wang, "Guided test case generation through ai enabled output space exploration," in *Proceedings of the 13th International Workshop on Automation of Software Test*, 2018.
- [125] J. Kim, R. Feldt, and S. Yoo, "Guiding deep learning system testing using surprise adequacy," 2018, arXiv preprint arXiv: 1808.08444.
- [126] L. Pulina and A. Tacchella, "An abstraction-refinement approach to verification of artificial neural networks," in *Proceedings of the 22nd International Conference on Computer Aided Verification*, 2010, pp. 243–257.
- [127] I. Goodfellow and N. Papernot, "The challenge of verification and testing of machine learning," 2017, <http://www.cleverhans.io/>.
- [128] G. Katz, C. Barrett, D. Dill, K. Julian, and M. Kochenderfer, "Reluplex: An efficient smt solver for verifying deep neural networks," in *Proceedings of the International Conference on Computer Aided Verification*, 2017, pp. 97–117.
- [129] L. D. Moura and N. Björner, "Satisfiability modulo theories: introduction and applications," vol. 54, no. 9, pp. 69–77, 2011.
- [130] J. P. Vielma, "Mixed integer linear programming formulation techniques," *SIAM Review*, vol. 57, no. 1, pp. 3–57, 2015.
- [131] V. Tjeng, K. Xiao, and R. Tedrake, "Evaluating robustness of neural networks with mixed integer programming," 2017, arXiv preprint arXiv: 1711.07356.
- [132] N. Carlini, G. Katz, C. Barrett, and D. L. Dill, "Ground-truth adversarial examples," 2017, arXiv preprint arXiv: 1709.10207.
- [133] O. Bastani, Y. Ioannou, L. Lampropoulos, D. Vytiniotis, A. Nori, and A. Criminisi, "Measuring neural net robustness with constraints," in *Advances in neural information processing systems*, 2016, pp. 2613–2621.
- [134] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana, "Formal security analysis of neural networks using symbolic intervals," 2018, arXiv preprint arXiv: 1804.10829.
- [135] R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to Interval Analysis*, 2009.
- [136] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, "Safety verification of deep neural networks," in *Proceedings of the International Conference on Computer Aided Verification*, 2017, pp. 3–29.
- [137] N. Narodytska, S. P. Kasiviswanathan, L. Ryzhyk, M. Sagiv, and T. Walsh, "Verifying properties of binarized deep neural networks," 2017, arXiv preprint arXiv: 1709.06662.
- [138] I. Hubara, D. Soudry, and R. E. Yaniv, "Binarized neural networks," in *Advances in neural information processing systems*, 2016, pp. 4107–4115.
- [139] E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, "Counterexample-guided abstraction refinement," in *Proceedings of International Conference on Computer Aided Verification*, 2000, pp. 154–169.
- [140] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [141] T. Gehr, M. Mirman, D. Drachler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev, "Ai2: Safety and robustness certification of neural networks with abstract interpretation," in *IEEE Symposium on Security and Privacy (SP)*, 2018.
- [142] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proceedings of the International Conference on Learning Representations*, 2018.
- [143] T.-W. Weng, H. Zhang, H. Chen, Z. Song, C.-J. Hsieh, D. Boning, I. S. Dhillon, and L. Daniel, "Towards fast computation of certified robustness for relu networks," 2018, arXiv preprint arXiv: 1804.09699.
- [144] M. Blohm, G. Jagfeld, E. Sood, X. Yu, and N. T. Vu, "Comparing attention-based convolutional and recurrent neural networks: Success and limitations in machine reading comprehension," in *Proceedings of the SIGNLL Conference on Computational Natural Language Learning*, 2018.
- [145] J. E. Hummel and K. J. Holyoak, "Relational reasoning in a neurally plausible cognitive architecture: An overview of the lisa project," vol. 14, no. 3, p. 153157, 2005.
- [146] M. Cheng, J. Yi, H. Zhang, P.-Y. Chen, and C.-J. Hsieh, "Seq2sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples," *arXiv preprint arXiv:1803.01128*, 2018.
- [147] R. Jia, A. Raghunathan, K. Göksel, and P. Liang, "Certified robustness to adversarial word substitutions," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.
- [148] K. Dvijotham, S. Gowal, R. Stanforth, R. Arandjelovic, B. O'Donoghue, J. Uesato, and P. Kohli, "Training verified learners with learned verifiers," *arXiv preprint arXiv:1805.10265*, 2018.
- [149] J. Zhang and X. Jiang, "Adversarial examples: Opportunities and challenges," 2018, arXiv preprint arXiv: 1809.04790.
- [150] W. Wang, R. Wang, L. Wang, and B. Tang, "Adversarial examples generation approach for tendency classification on chinese texts," *Ruan Jian Xue Bao/Journal of Software*, pp. 1–14, 2019.
- [151] D. Goodman, "Advbox: a toolbox to generate adversarial examples that fool neural networks," 2019. [Online]. Available: <https://github.com/baidu/AdvBox>
- [152] N. Papernot, I. Goodfellow, R. Sheatsley, R. Feinman, and P. McDaniel, "cleverhans v1.0.0: an adversarial machine learning library," *arXiv preprint arXiv:1610.00768*, 2016.
- [153] B. Laughlin, C. Collins, K. Sankaranarayanan, and K. El-Khatib, "A visual analytics framework for adversarial text generation," *arXiv preprint arXiv:1909.11202*, 2019.
- [154] F. Suya, J. Chi, D. Evans, and Y. Tian, "Hybrid batch attacks: Finding black-box adversarial examples with limited queries," in *Proceedings of the 29th USENIX Security Symposium*, 2020.
- [155] F. Menet, P. Berthier, J. M. Fernandez, and M. Gagnon, "Spartan networks: Self-feature-squeezing neural networks for increased robustness in adversarial settings," in *CCS '18 Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 2246–2248.
- [156] S. Kariyappa and M. K. Qureshi, "Improving adversarial robustness of ensembles with diversity training," 2019, arXiv preprint arXiv: 1901.09981.
- [157] K. Nar, O. Ocal, S. S. Sastry, and K. Ramchandran, "Cross-entropy loss and low-rank features have responsibility for adversarial examples," 2019, arXiv preprint arXiv: 1901.08360.
- [158] CloudFlower, "Twitter gender classification dataset," 2013. [Online]. Available: <https://www.kaggle.com/crowdflower/twitter-user-gender-classification>

- [159] R. Johnson and T. Zhang, “Effective use of word order for text categorization with convolutional neural networks,” in *Proceedings of the North American Chapter of the Association for Computational Linguistics Human Language Technologies (NAACL HLT)*, 2015.
- [160] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “Squad: 100,000+ questions for machine comprehension of text,” in *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*, 2016.
- [161] A. Williams, N. Nangia, and S. R. Bowman, “A broad-coverage challenge corpus for sentence understanding through inference,” in *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.
- [162] M. Tapaswi, Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun, and S. Fidler, “Movieqa: Understanding stories in movies through question-answering,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

IX. APPENDIX A

TABLE VI: Instances of some methods in section 3

Method	Instance of adversarial example	type of modification
Alzantot et al. [57]	premise: A runner wearing purple strives for the finish line. original: A runner wants to head for the finish line. 86% Entailment modified: A racer wants to head for the finish line. 43% Contradiction	word-level
DeepWordBug [61]	original: This film has a special place in my heart. positive modified: This film has a special plcae in my herat . negative	multi
HotFlip [64]	original: South Africas historic Soweto township marks its 100th birthday on Tuesday in a mood of optimism. 57% World modified: South Africas historic Soweto township marks its 100th birthday on Tuesday in a moo P of optimism. 95% Sci/Tech	char-level
iAdv-Text [65]	original: The essence of this film falls on judgments by police officers who fortunately ethical and moral men act on situations within situations in a city with a super abundance of violence and killing Good compound interacting story lines and above average characterizations. positive modified: The essence from THIS film falls on judgments by police officers who fortunately ethical and moral men act on situations within situations in a city with a super abundance of violence and killing Good compound interacting story lines and above average characterizations. negative	char-level
Text-fool [66]	original: The Old Harbor Reservation Parkways are three historic roads in the Old Harbor area of Boston. They are part of the Boston parkway system designed by Frederick Law Olmsted. They include all of William J. Day Boulevard running from Castle Island to Kosciuszko Circle along Pleasure Bay and the Old Harbor shore. The part of Columbia Road from its northeastern end at Farragut Road west to Pacuska Circle (formerly called Preble Circle). 87.3% Building modified: The Old Harbor Reservation Parkways are three historie roads in the Old Harbor area of Boston. Some exhibitions of Navy aircrafts were held here. They are part of the Boston parkway system designed by Frederick Law Olmsted. They include all of William J. Day Boulevard running from Cast l e Island to Kosciuszko Circle along Pleasure Bay and the Old Harbor shore. The part of Columbia Road from its northeastern end at Farragut Road west to Pacuska Circle formerly called Preble Circle. 95.7% Means of Transportation	multi
TextBugger [67]	original: reason why requesting i want to report something so can ips report stuff, or can only registered users can? if only registered users can, then i 'll request an account and it 's just not fair that i cannot edit because of this anon block shit c'mon, fucking hell. Toxic modified: reason why requesting i want to report something so can ips report stuff, or can only registered users can? if only registered users can, then i 'll request an account and it 's just not fair that i cannot edit because of this anon block shiti c'mon, fucking helled . Non-toxic	multi
Samanta et al. [68]	original: A sprawling, overambitious, plotless comedy that has no dramatic center. It was probably intended to have an epic vision and a surrealistic flair (at least in some episodes), but the separate stories are never elevated into a meaningful whole, and the laughs are few and far between. Amusing ending though. negative modified: A sprawling, overambitious, plotless funny that has no dramatic center. It was probably intended to have an epic vision and a surrealistic flair (at least in some episodes), but the separate stories are never elevated into a greatly whole, and the laughs are little and far between amusing ending though. positive	word-level
Gong et al. [69]	original: One of those TV films you saw in the seventies that scared the hell out of you when you were a kid but still gives you an eerie feeling. No great actors or expensive production but everytime that phone rings Label 0 modified: One of those TV films you saw in the seventies that scared the hell out of you when you were a kid but not gives you an considered unnerving . No great actors and/or expensive production but everytime that phone rings Label 1	word-level
Papernot et al. [71]	original: I wouldn't rent this one even on dollar rental night. negative modified: Excellent wouldn't rent this one even on dollar rental night. positive	word-level

X. APPENDIX B

TABLE VII: Other datasets used in research works

dataset	application in the work	task	source
Enron Spam	[61]	Spam E-mail Detection	-
Twitter dataset	[68]	gender prediction	[158]
Elec ¹	[65]	sentiment analysis	[159]
RCV1 ¹	[65]	classification	[159]
FCE-public	[65]	grammatical error detection	-
Stanford Sentiment Treebank	[64], [93]	sentiment analysis	-
Stanford Question Answering Dataset (SQuAD) ²	[87]	attacking reading system	[160]
MultiNLI ³	[70]	attacking Natural Language Inference system	[161]
MovieQA dataset ⁴	[144]	attacking reading system	[162]
Customer review dataset(CR) ⁵	[66]	sentiment analysis	-
Reuters ⁶	[69]	classification	-

¹ http://riejohnson.com/cnn_data.html² <https://stanford-qa.com>³ <http://www.nyu.edu/projects/bowman/multinli/>⁴ <http://movieqa.cs.toronto.edu/leaderboard/>⁵ <https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html>⁶ <http://www.daviddlewis.com/resources/testcollections/reuters21578/>